precise forms disclosed. Numerous modifications, adaptations, and uses thereof will be apparent to those skilled in the art without departing from the scope of the disclosure. For instance, examples described herein can be combined together to yield still further examples.

1. A system comprising:

a processor; and

a memory including instructions that are executable by the processor for causing the processor to:

receive an input from a client device, the input specifying a target software item and specifying a characteristic of a computing environment in which the target software item is to be executed;

generate software-stack candidates for the target software item by performing a search using a search algorithm configured to recursively analyze direct and indirect dependencies of the target software item, the software-stack candidates having unique configurations of software components including the target software item and dependencies of the target software item;

determine a respective score for each software-stack candidate of the software-stack candidates based on the characteristic of the computing environment and a unique configuration of software components forming the software-stack candidate;

select a particular software-stack candidate from the software-stack candidates as a recommended software-stack, based on the respective score for the particular software-stack candidate having a predefined attribute; and

transmit an output to the client device indicating the recommended software-stack, wherein the client device is operable to install the recommended software-stack in the computing environment.

2. The system of claim **1**, wherein the software-stack candidates include different versions of the target software item and different versions of dependencies of the target software item.

3. The system of claim **1**, wherein the characteristic of the computing environment is a hardware characteristic or a software characteristic of the computing environment.

4. The system of claim **1**, wherein the memory further includes instructions that are executable by the processor for causing the processor to generate the respective score for each respective software-stack candidate among the software-stack candidates using a scoring function configured to account for the characteristic of the computing environment and (i) a performance property of a first software component in the respective software-stack candidate or (ii) a security property of a second software component in the respective software-stack candidate.

5. The system of claim **4**, wherein the memory further includes instructions that are executable by the processor for causing the processor to determine the performance property and the security property based on information in a predefined database, the information including relationships between (i) software components usable in software stacks, (ii) performance characteristics of the software components, and (iii) security characteristics of the software components.

6. (canceled)

7. The system of claim **1**, wherein the search is a heuristic search, and wherein the memory further includes instructions that are executable by the processor for causing the processor to determine the software-stack candidates by performing the search using a heuristic search algorithm.

8. The system of claim **7**, wherein the heuristic search algorithm is a Monte Carlo tree-search algorithm or a temporal-difference learning algorithm.

9. The system of claim **7**, wherein the heuristic search algorithm is configured to recursively analyze the direct and indirect dependencies of the target software item to expand a search space associated with the heuristic search.

10. The system of claim **1**, wherein the memory further includes instructions that are executable by the processor for causing the processor to:

determine a recommended file to include in the recommended software-stack, wherein the recommended file is different from the target software item and is not a dependency of the target software item, and wherein the recommended file is configured to improve a performance characteristic or a security characteristic of the recommended software-stack; and

include the recommended file in the recommended software-stack.

11. A method comprising:

receiving, by a processor and from a client device, an input specifying a target software item and specifying a characteristic of a computing environment in which the target software item is to be executed;

generating, by the processor, software-stack candidates for the target software item by performing a search using an algorithm configured to recursively analyze direct and indirect dependencies of the target software item, the software-stack candidates having unique configurations of software components including the target software item and dependencies of the target software item;

determining, by the processor, a respective score for each software-stack candidate of the software-stack candidates based on the characteristic of the computing environment and a unique configuration of software components forming the software-stack candidate;

selecting, by the processor, a particular software-stack candidate from the software-stack candidates as a recommended software-stack, based on the respective score for the particular software-stack candidate having a predefined attribute; and

transmitting, by the processor, an output to the client device indicating the recommended software-stack, wherein the client device is operable to install the recommended software-stack in the computing environment.

12. The method of claim **11**, wherein the software-stack candidates include different versions of the target software item and different versions of dependencies of the target software item.

13. (canceled)

14. The method of claim **11**, further comprising generating the respective score for each respective software-stack candidate among the software-stack candidates using a scoring function configured to account for (i) the characteristic of the computing environment, (ii) a performance property of a first software component in the respective software-stack candidate, and (iii) a security property of a second software component in the respective software-stack candidate.